DAFTAR ISI

BAB I

PENDAHULUAN

1.1 Latar Belakang

Seiring dengan berkembangnya teknologi saat ini, manusia sudah dipermudahkan dengan segala sesuatu yang serba praktis dan mudah. Salah satunya adalah dalam pengiriman surat menyurat sekarang dapat dilakukan via *E-mail* dan *chat* menggunakan internet. Seiring berkembang nya teknologi ini makin banyak benda-benda yang digunakan manusia digantikan menjadi digital, salah satunya adalah peta atau *map*. Peta tradisional yang terbuat dari kertas dengan gambaran wilayah diatasnya kini sudah tergantikan oleh *Google Maps* yang lebih praktis dan memiliki jangkauan lebih luas dibandingkan peta tradisional.

Google Maps ini dapat digunakan melalui *smartphone* atau melalui *desktop*, sehingga dalam penggunaan *Google Maps* ini sangatlah praktis. Pemanfaatan *Google Maps* ini biasanya digunakan sebagai navigasi dalam mencari suatu tempat dengan memanfaatkan sistem *GPS* nya untuk menuntun *user* ke jalan yang benar.

Penggunaan API *Google Maps* saat ini sudah banyak dimanfaatkan dalam beberapa bidang teknologi contoh nya sebagai sarana belajar navigasi mahasiswa [1], Pemetaan dan pemberdayaan pariwisata desa [2], atau bahkan dapat diterapkan pada aplikasi monitoring lokasi anak [3] dan dapat digunakan untuk membantu polres mengelola laporan kriminal [4], membantu mencari situasi genting [5] atau dapat membantu mengetahui jadwal *bus stop* pada halte bus[6].

Penulis menemukan permasalahan dalam pemesanan makanan dalam jumlah pemesanan dalam jumlah banyak terutama pada kantin , foodcourt ataupun

restoran, di mana pengantar makanan merasa kesulitan untuk menemukan di mana posisi pemesan makanan duduk di dalam kondisi ramai.

Berdasarkan masalah tersebut, penulis ingin mengembangkan penggunaan API *Google Maps* untuk membuat aplikasi pemesanan makanan pada restoran dengan menggunakan *Google Maps* API dengan metode *Geocode* di mana aplikasi akan secara otomatis mengenali meja yang diduduki oleh *costumer* .Aplikasi ini akan mempermudah pengantar makanan mengenali meja *costumer* berdasarkan *order* yang dibuat.

Dalam penelitian ini penulis ingin menerapkan system mapping menggunakan koordinat API *Google Maps* yang berbasis *longitude* dan *latitude* untuk diterapkan di aplikasi sederhana untuk menentukan lokasi meja makan yang diduduki *costumer* di sebuah restoran di daerah Palgading, Ngaglik, Yogyakarta di mana ketika *costumer* memesan makanan melalui aplikasi yang dibuat maka secara otomatis akan terdaftar lokasi meja yang diduduki.

Penulis pertama-tama akan melakukan mapping di objek penelitian dengan mencari titik koordinat *latitude* dan *longitude* di sekeliling meja yang digunakan. Titik koordinat akan dicatat berdasarkan garis koordinat *latitude* dan *longitude* tiap meja lalu akan di buat logika di mana jika *costumer* (koordinat yang terdeteksi) duduk ditengah koordinat yang sudah diterapkan maka akan terdeteksi secara otomatis *costumer* sedang berada di meja yang terdaftar, sehingga aplikasi ini dalam penempatan koordinat meja makan ditentukan secara statis.

Setelah diterapkan, maka penulis akan menghitung ke akurasian dalam pendeteksian koordinat yang dilakukan oleh aplikasi lalu akan dinilai apakah

metode ini layak untuk diterapkan pada aplikasi pemesanan ini atau tidak dan apakah cocok dengan objek yang digunakan.

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah yang diuraikan sebelumnya, maka dapat di rumuskan masalah sebagai berikut :

1. Apakah penggunaan metode *geocode* dalam pengambilan koordinat *customer* berdasarkan *latitude* dan *longitude* didapat secara akurat ?
2. Apakah penggunaan metode *geocode* pada pemesanan makanan akan lebih cepat terselesaikan?
3. Apakah metode *geocode* cocok digunakan di objek yang diteliti ?
4. Apakah dengan metode *geocode* proses pemesanan makanan akan menjadi praktis atau tidak?

1.3 Batasan Masalah

Penelitian dilakukan di objek berupa restoran yang menyediakan makanan-makanan seperti soto,bakso dan mi ayam. Batasan utama dari penelitian ini adalah:

1. Fitur-fitur yang nantinya digunakan pada aplikasi adalah fitur-fitur dasar yang digunakan untuk pemesanan makanan di sekitar restoran.
2. Ketergantungan terhadap sinyal dan cuaca dapat berpengaruh besar dalam pendeteksian koordinat user.
3. Besar kecil ukuran meja dan kursi lokasi user makan juga mempengaruhi dalam penetapan lokasi koordinat meja. Semakin besar lokasi makan user semakin tinggi akurasi dalam penemuan koordinat user.

1.4 Tujuan Penelitian

Tujuan dalam penelitian ini adalah sebagai berikut :

1. Untuk mengetahui keakurasian koordinat yang dideteksi menggunakan API *Google Maps* dalam menentukan lokasi.
2. Untuk mempercepat proses pemesanan makanan.
3. Untuk mengetahui apakah objek yang di teliti cocok menggunakan dengan metode ini.
4. Untuk mengetahui apakah metode ini cocok digunakan dalam aplikasi pemesanan makanan.

1.5 Manfaat Penelitian

1.5.1 Manfaat Bagi Objek

Manfaat untuk objek dari penelitian ini adalah diharapkan penelitian ini dapat membantu mempercepat proses pemesanan makanan pada sebuah penyedia makanan, rumah makan atau *foodcourt* sehingga mengurangi waktu berjalan ke kasir untuk memesan makanan. Juga dapat mempermudah pengantar makanan dalam mengantarkan makanan kepada *costumer* tanpa harus memastikan pesanan kepada *costumer* yang sedang duduk.

1.5.2 Manfaat Bagi Peneliti

Manfaat untuk peneliti dari penelitian ini adalah sebagai referensi bagi *developer* yang akan menggunakan metode ini dalam menentukan koordinat yang dicari dalam ruang lingkup yang tidak terlalu besar. Dan dapat bermanfaat untuk mempermudah dan mempercepat sistem pelayanan pemesanan makanan atau sistem berbasis location mapping lain.

1.6 Sistematika Penulisan

Pada penelitian ini, sistematika penulisan dilakukan dengan mengelompokkan materi-materi menjadi beberapa bab berikut:

Bab I : Pendahuluan : Pada bab ini, dijelaskan informasi umum dari penelitan ini, yaitu : latar belakang penelitian, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metode penelitian dan sistematika penulisan.

Bab II : **Dasar Teori** : Pada bab ini berisi landasan teori, analisa penelitian terdahulu, dan hipotesis yang berhubungan dengan penelitian yang dilakukan

Bab III : **Metodologi Penelitian** : Pada bab ini, berisikan perancangan sistem, bagaimana cara pengambilan dan pengolahan data pada penelitian ini. Juga dijelaskan Analisa dan Desain dari penelitian ini menggunakan metode *SDLC Waterfall*.

Bab IV : **Implementasi** : Pada bab ini, berisikan tentang bagaimana hasil dari penerapan aplikasi pada objek, pengimplementasian metode yang digunakan dan pengujian aplikasi.

Bab V : **Penutup** : Pada bab ini, berisikan kesimpulan dari penelitian serta saran-saran dari peneliti.

BAB II

LANDASAN TEORI

2.1 Tinjauan Pustaka

2.1.1 Integrasi Aplikasi Web dan Android

Percobaan yang dilakukan oleh Jos Forman Tompoh (Tompoh et al., 2016) memanfaatkan *framework cordova* dan *framework ionic* sehingga dapat diakses melalui web dan android secara server lokal dengan menggunakan XAMPP. Dengan menggunakan web-services *client*, user android dan *client* admin pada web dapat berintegrasi dalam sistem pemesanan.

Pada Aplikasi yang dirancang oleh Wismarini dan Prihandono (Wismarini & Prihandono, 2020), melakukan integrasi aplikasi android dengan web services menggunakan library *volley* yang memanfaatkan REST Api sebagai perantara dengan web service nya.Sistem yang diterapkan juga masih menggunakan web server secara lokal.

2.1.2 Penerapan Metode Geocode pada Aplikasi Android

Dalam Aplikasi yang dikembangkan oleh Sandheep S, Harry John, Harikumar A dan Vinitha Panicker J (Sandheep et al., 2017), diterapkan metode *Reverse Geocoding* untuk mengubah data koordinat yang dikoleksi dengan metode *Crowdsourcing* menjadi lokasi *bus stop* terdekat pada Google Map berdasarkan koordinat yang didapat oleh user setelah turun atau naik bus. Sistem ini akan berkembang lebih cepat jika semakin banyak

user yang menggunakan, dikarenakan aplikasi akan menerima lebih banyak data dari *Crowdsourcing* dan akan lebih efektif.

Pada perancangan dan pembuatan aplikasi *Emergency Search* yang dirancang oleh Anisha Ginjala (Ginjala, 2015), Menggunakan Twitter API untuk mendapatkan postingan atau *tweet* tentang keadaan darurat, lalu ditampilkan pada *Google Maps* melalui aplikasi ini.

User juga dapat memposting atau menuliskan suatu keadaan darurat melalui aplikasi ini dan menuliskan lokasi kejadian, jika user tidak menuliskan lokasi maka secara otomatis lokasi akan tertulis menggunakan metode *Reverse Geocoding* dengan mengubah titik *Latitude* dan *Longitude* menjadi lokasi suatu tempat sehingga dapat dibaca oleh user lain nya.

2.2 Landasan Teori

2.2.1 Peta

Peta merupakan penyajian grafis dari permukaan bumi dalam skala tertentu dan digambarkan pada bidang datar melalui sistem proyeksi peta dengan menggunakan symbol-simbol tertentu sebagai perwakilan dari objek-objek spasial yang berada di permukaan bumi seperti gunung, jalan, hutan dll[7].

2.2.2 Google Maps

Google Maps adalah peta digital yang dapat digunakan untuk melihat suatu daerah[7] yang dikembangkan oleh Google dan diluncurkan secara publik pada bulan Februari tahun 2005 , bermula dari program *desktop* dua orang bersaudara dari Denmark yaitu Lars Rasmussen yang ingin menyaingi program digital mapping yang sudah ada seperti MapQuest, lalu Google mendukung startup dari Rasmussens bersaudara di tahun 2004[8].

2.2.3 Google Maps API

Google Maps API adalah suatu library berbentuk JavaScript [7] yang disediakan oleh Google melalui *google play services library* yang dapat didownload secara eksternal dari situs android developer atau dari *android SDK manager*.Sehingga aplikasi android dapat mengintegrasi fitur-fitur Google Maps untuk dimanfaatkan fungsi-fungsi nya untuk digunakan pada berbagai bidang [9].

2.2.4 Global Positioning System (GPS)

Fitur yang sering digunakan oleh masyarakat dalam penggunaan *Google Maps* saat ini adalah GPS atau dapat dikenal dengan *Global Positioning System*, GPS merupakan sistem navigasi dan penentu lokasi berbasis satelit dengan tingkat ketelitian tinggi [3]. GPS telah dikembangkan dalam bentuk smartphone sehingga penggunaannya lebih mudah sehingga datanya dapat digunakan untuk mengambil data koordinat dari masing-masing pengguna smartphone [3].

2.2.5 Location-Based Services (LBS)

LBS adalah layanan berbasis lokasi, yaitu sebuah layanan berbasis internet yang berfungsi untuk mencari lokasi dengan berbasis GPS. Map dan layanan berbasis lokasi menggunakan lintang bujur bumi (*longitude* dan *latitude*). Android telah menyediakan *geocoder* yang dapat mendukung *forward* dan *reverse geocoding*[10].

Dengan *geocode* nilai lintang bujur (*longitude* dan *latitude*) dapat dikonversikan menjadi alamat yang dapat dikenali secara *plain text*[10].

2.2.6 Geocoding

Geocoding adalah proses konversi deskripsi lokasi berbasis *text-based* menjadi sebuah nilai koordinat. Namun proses *geocoding* ini masih memiliki kesalahan spasial yang berpengaruh terhadap *output* sehingga dapat berpengaruh dalam kevalidasi dan keakurasian pengukuran lokasi dan jarak pada *geocoding* itu sendiri[11].

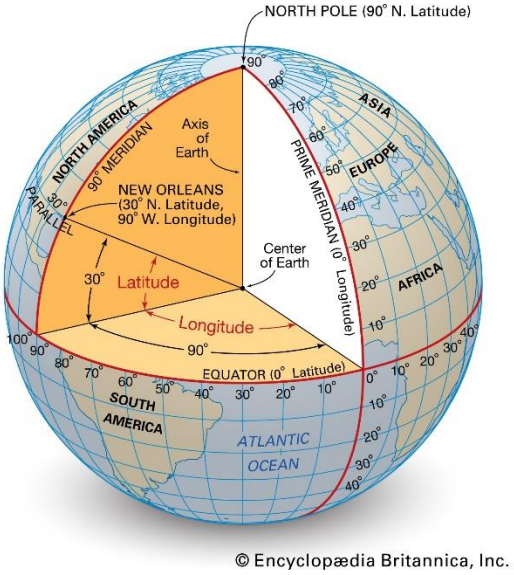
Geocoding dapat dimanfaatkan sebagai simple data analysis sampai keperluan bisinis dan manajemen kostumer[12].

Geocoding memiliki dua metode yaitu *Forward Geocoding* dan *Reverse Geocoding*. *Forward Geocoding* adalah proses konversi suatu alamat dari *plain text* menjadi koordinat geografik[13]. Sedangkan *Reverse Geocoding* adalah proses konversi koordinat geografik menjadi alamat secara *plain text* yang dapat dibaca oleh manusia[14].

Dalam penelitian ini, penulis akan menggunakan metode *Reverse Geocoding* yang mengkonversi koordinat *latitude* dan *longitude* melalui aplikasi dan dikonversi menjadi sebuah informasi berupa *plain text* yang dapat dibaca oleh user berupa nomor meja. Sehingga secara otomatis jika user duduk di salah satu lokasi koordinat yang sudah di tetapkan, maka secara otomatis aplikasi mengambil lokasi user saat ini dan menjalankan proses *Reverse Geocoding* dan mengubah lokasi koordinat menjadi sebuah informasi berupa nomor meja yang ditempati.

2.2.7 Longitude dan Latitude

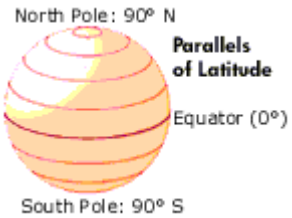
Dalam penentuan koordinat lokasi di GPS sendiri ditentukan oleh *longitude, latitude.Latitude* dan *longitude* dihitung berdasarkan sudut yang bertumpu dari inti bumi[15].



Gambar 5.1 Ilustrasi inti bumi terhadap longitude dan latitude

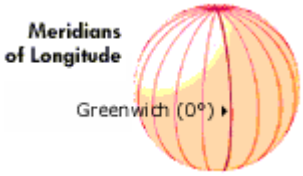
Latitude adalah garis yang melintang secara horizontal sejajar dengan garis *equator* yang memutari bumi dan berlawanan dengan garis meridian utama .Dengan batas + dan – berdasarkan jauh dekat nya garis dengan garis *equator* bumi. Dari kutub utara ke *equator* adalah garis melintang positif (+) dan garis melintang setelah *equator* sampai ke kutub selatan adalah garis melintang negatif (-)[16].

Garis *equator* adalah titik pusat dari *latitude* sehingga garis *equator* memiliki 0 derajat latitude. Sudut *latitude* akan lebih besar semakin garis *latitude* menjauhi garis pusat *equator* sehingga kutub utara dan kutub selatan memiliki sudut *latitude* sebesar 90 derajat.

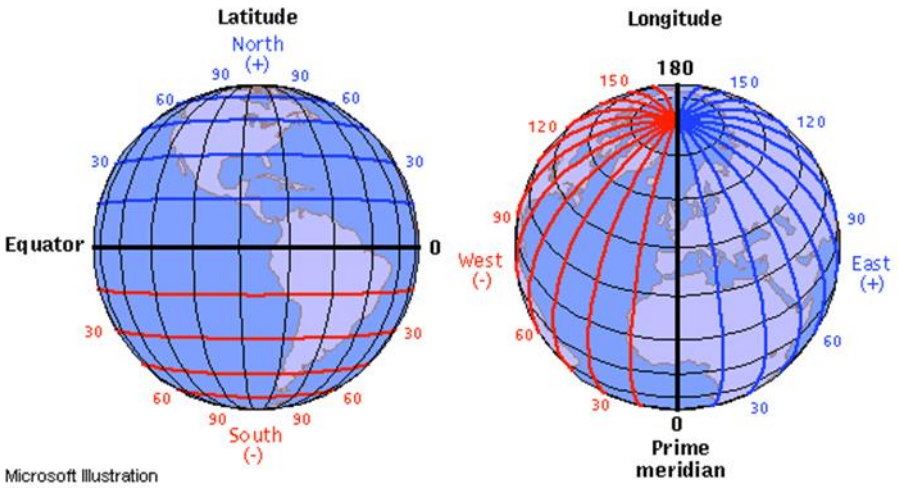


Gambar 5.2 Ilustrasi garis melintang latitude terhadap garis equator

Longitude juga disebut sebagai *meridian* [17] adalah garis melintang secara vertikal yang sejajar dengan garis meridian utama dan berlawanan dengan garis *equator* bumi. Dengan batas + dan – berdasarkan jauh dekat nya dengan garis meridian utama. Wilayah Greenwich di Inggris adalah titik pusat dari *longitude* atau memiliki *longitude* 0 derajat[18] sehingga perhitungan dihitung dari arah wilayah Greenwich ke kanan adalah kutub positif (+) dari *longitude* dan dari wilayah Greenwich kekiri adalah kutub negatif (-) dari *longitude*[16].



Gambar 5.3 Ilustrasi garis vertikal longitude terhadap garis meridian utama



Gambar 5.4 Ilustrasi perbandingan dari garis latitude dan longitude

2.2.8 Android

Android adalah sebuah sistem operasi untuk perangkat *mobile* berbasis linux yang dapat mencakup sistem operasi, middleware dan aplikasi[19].

Karena android berbasis linux maka android termasuk *open source* sehingga siapapun dapat memodifikasi sistem operasi nya secara bebas, berbeda dengan iOS yang bersifat closed source[20]. Bahasa pemrograman yang digunakan pada android adalah Java[21].

2.2.9 Aplikasi

Aplikasi adalah program siap pakai yang dapat digunakan untuk menjalankan perintah-perintah dari user atau pengguna aplikasi tersebut dengan tujuan untuk mendapatkan hasil yang lebih akurat sesuai dengan tujuan pembuatan aplikasi tersebut.

Secara umum, pengertian aplikassi adalah alat terapan yang difungsikan secara khusus dan terpadu sesai dengan kemampuan yang dimilikinya, aplikasi merupakan suatu perangkat komputer yang siap pakai bagi user[10].

2.2.10 Bahasa Pemrograman PHP

PHP atau dapat dikenal sebagai *PHP Hypertext Processor* digunakan sebagai bahasa script dalam pengembangan web yang dimasukan pada dokumen HTML. Penggunaan PHP memungkinkan web berkomunikasi dengan aplikasi desktop maupun android menggunakan fungsi API (*Aplication Programming Interface*) sehingga aplikasi dapat menyimpan sebuah database pada MySQL secara online.

2.2.11 Database MySQL

MySQL adalah sistem database open source yang paling populer di seluruh dunia dikarenakan MySQL adalah open source sehingga semua orang dapat menggunakan dan mengembangkan MySQL untuk segala kebutuhan secara gratis.

MySQL adalah sistem database yang paling sering digunakan aplikasi berbasis web[7] dan dapat berkomunikasi satu sama lain dengan aplikasi dengan perantara API.

2.2.12 Unified Modeling Language (UML)

Unified Modeling Language atau yang dapat disebut UML adalah satu metode pemodelan visual yang digunakan untuk perancangan dan pembuatan sebuah aplikasi atau *software* yang berorientasikan pada objek atau *Object Oriented Software*. [22]

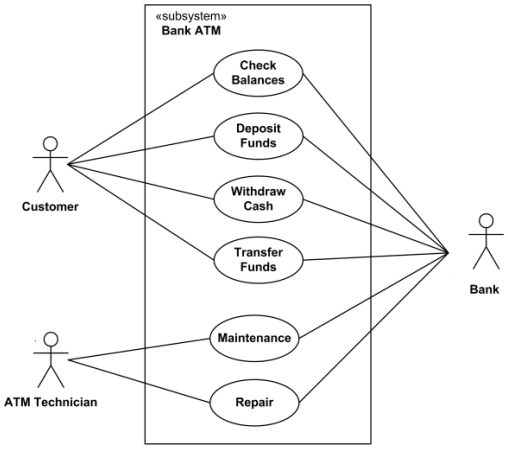
UML merupakan sebuah standar penulisan yang memiliki rancangan-rancangan yang berisi bisnis proses, penulisan kelas-kelas dalam sebuah bahasa yang lebih spesifik dan detil.[22]

UML memiliki beberapa diagram yang sering digunakan dalam pengembangan sebuah sistem, yaitu :

2.2.12.1 *Use Case Diagram*

Suatu sistem *Use Case Diagram* adalah salah satu jenis dari diagram UML yang menggambarkan hubungan interaksi antara sistem dan aktor(pengguna). *Use Case Diagram* dapat

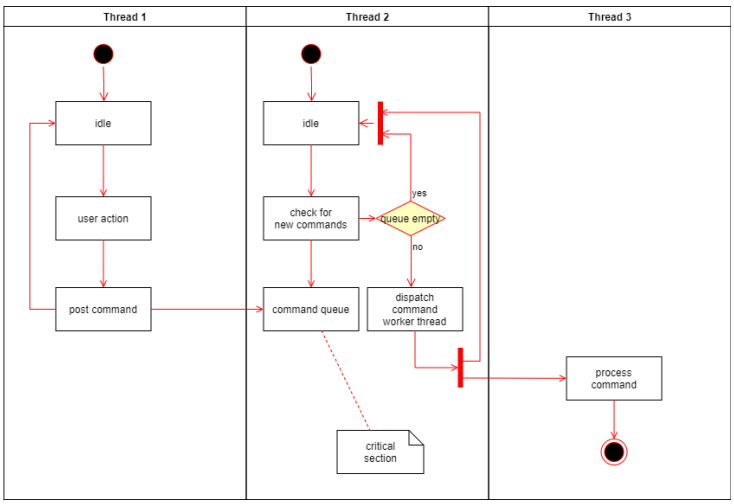
mendeskripsikan tipe-tipe interaksi antara pengguna sistem dengan sistem yang digunakan.[23]



Gambar 5.5 Contoh Use Case Diagram

2.2.12.2 *Activity diagram*

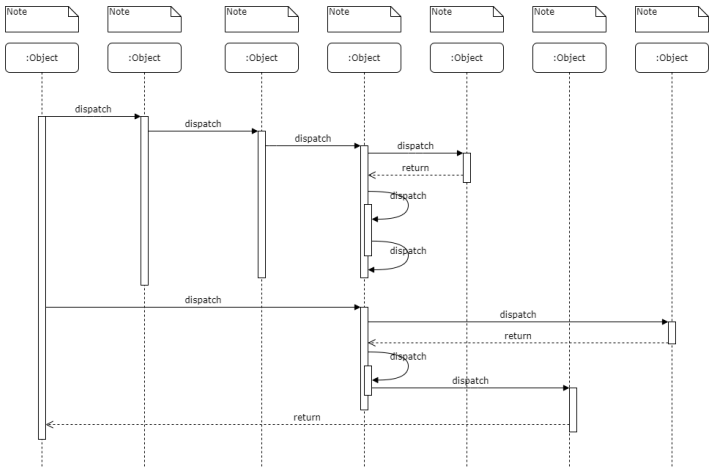
merupakan sebuah diagram yang dapat menggambarkan model berbagai proses yang terjadi pada sistem. Seperti runtutan proses berjalannya suatu sistem dan digambarkan secara vertikal.[23]



Gambar 5.6 Contoh Activity Diagram

2.2.12.3 *Sequence Diagram*

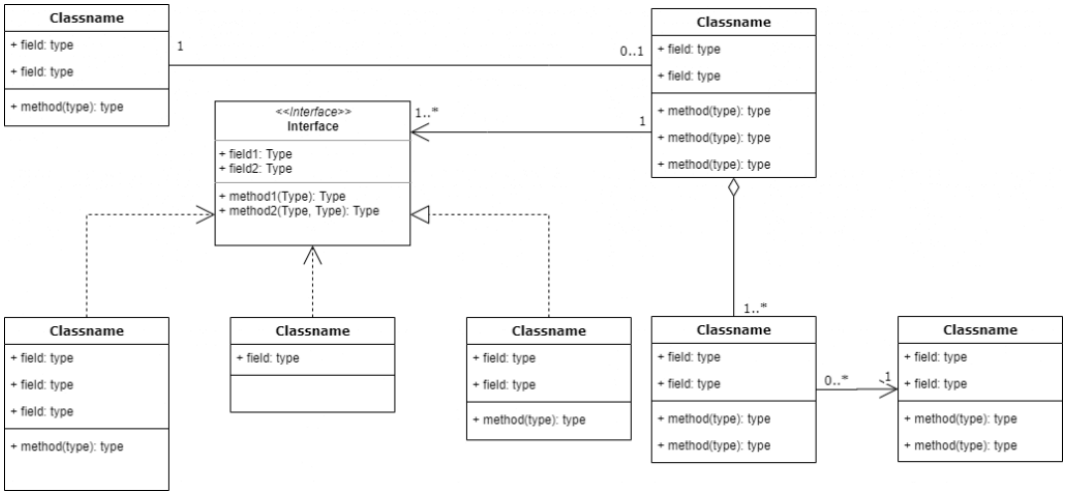
Sequence Diagram menggambarkan Interaksi antar objek didalam dan di sekitar sistem yang berupa pesan yang digambarkan terhadap waktu.[22]



Gambar 5.7 Contoh Sequence Diagram

2.2.12.4 *Class Diagram*

Class Diagram merupakan gambaran struktur dan deskripsi dari suatu class, package dan objek yang saling berhubungan seperti diantaranya pewarisan, asosiasi dan lainnya.[22]



Gambar 5.8 Contoh Class Diagram

2.2.13 JavaScript Object Notation (JSON)

JSON adalah format pertukaran data yang ringan dan dapat lebih mudah dimengerti. JSON juga lebih mudah diproses oleh mesin atau software[24].

2.2.14 Application Programming Interface (API)

API adalah software perantara yang berfungsi untuk melakukan komunikasi antar dua aplikasi[25]. Contoh nya adalah aplikasi *mobile* mengirimkan perintah input yang digunakan untuk menampilkan data dari database dikirimkan melalui API, lalu API meneruskan permintaan input tersebut ke Database, lalu dari Database mengirimkan kembali output yang diminta oleh aplikasi *mobile* melalui API berupa *JSON text* dan ditampilkan oleh aplikasi *mobile.*

2.2.15 REST API

REST API adalah salah satu teknologi API yang sering digunakan ketika akan mengembangkan sebuah WEB API. REST API sendiri memiliki sifat *stateless*, yaitu dimana setiapkali *request* harus menyertakan semua data dan parameter dengan lengkap dan benar ketika mengakses suatu *endpoint*.[26]

Pada Arsitektur REST,REST server menyediakan *resources* dan REST client akan mengakses dan menampilkan *resource* tersebut untuk pengguna selanjutnya. *Resource* direpresentasikan dalam bentuk format teks, seperti JSON atau XML.[26]

REST memiliki standarisasi dalam pemakaian yaitu URL dan HTTP *method.* HTTP *method* diggunakan untuk mengetahui fungsi dari URL yang diakses, sehingga mempermudah dalam penulisan URL. [26]

Jenis HTTP *method* yang sering digunakan adalah :

1. **GET**, digunakan untuk membaca sebuah *record* atau data.
2. **POST**, digunakan untuk menambahkan *record* atau data.
3. **PUT**, digunakan untuk mengubah semua *field* dalam sebuah *record* atau data.
4. **PATCH**, digunakan untuk mengubah beberapa *field* dalam sebuah *record* atau data.
5. **DELETE**, digunakan untuk menghapus sebuah *record* atau data.

2.2.16 Volley

Volley adalah HTTP *library* pada *android studio* [27] untuk mempermudah dan kecepatan proses koneksi aplikassi android dengan jaringan.[28]

Pada penelitian ini penulis menggunakan *library Volley* untuk menjalin komunikasi antara aplikasi android dengan *REST API* yang digunakan.

Volley digunakan untuk mengirim input dari aplikasi menuju *REST API* yang akan diteruskan untuk membuat data atau mengambil data pada database.

Data yang dikirimkan oleh Volley ke *REST API* berupa parameter yang dibutuhkan yang nantinya akan di olah di *Function* yang berada di *REST API*.

Volley akan menerima output yang dikirim kan dari *REST API* berupa format teks JSON.

2.2.17 Entity Relationship Diagram (ERD)

ERD adalah suatu diagram untuk menggambarkan desain konseptual dari model konseptual suatu basis data relasional[29]. ERD merupakan suatu model untuk menjelaskan hubungan antar data dalam basis data berdasarkan objek-objek dasar data yang mempunyai hubungan antar relasi satu sama lain[7].

ERD memiliki hubungan erat dengan *Data Flow Diagram* (DFD) untuk menampilkan sebuah data yang disimpan. Yang bertujuan untuk

memvisualisasikan proses data yang dapat saling terhubung dan dapat menkonstruksi data relasional[30]. ERD memiliki beberapa komponen yang dipakai, yaitu :

2.2.17.1 Entitas

Entitas merupakan kumpulan dari objek-objek yang dapat teridentifikasi secara Unik. Di dalam ERD, entitas dilambangkan dengan persegi panjang. Dan entitas lemah akan digambarkan dengan bentuk persegi panjang kecil didalam persegi panjang yang besar[30].

2.2.17.2 Atribut

Setiap entitas memiliki bermacam macam atribut yang berfungsi untuk mendeskripsikan karakteristik dari entitas tersebut.Penggunaan atribut kunci atau *key* adalah pembeda dari entitas dan atribut yang diwakili dengan simbol elips[30]. Macam-macam atribut adalah sebagai berikut :

2.2.17.2.1 Atribut Kunci

Atribut kunci adalah atribut yang digunakan untuk menentukan data yang bersifat unik. Biasanya data dari atribut *key* berbentuk angka seperti NIK, NIM ,dan lain sebagainya[30].

2.2.17.2.2 Atribut Simpel

Atribut simpel adalah atribut yang tidak dapat dipecah lagi dan bernilai tunggal. Seperti alamat rumah, kantor, tahun terbit jurnal dan lain sebagainya[30].

2.2.17.2.3 Atribut Multinilai (*Multivalue*)

Atribut multi nilai adalah atribut yang memiliki sekelompok nilai untuk tiap entitasnya. Seperti kumpulan namaa pengarang dalam novel[30].

2.2.17.2.4 Atribut Gabungan (*Composite*)

Atribut gabungan adalah atribut yang berasal dari susunan atribut yang lebih kecil pada artian tertentu. Seperti data yang berhubungan pada nama lengkap yaitu nama depan, tengah dan nama belakang[30].

2.2.17.2.5 Atribut Derivatif

Atribut Derivatif adalah atribut yang berasal dari atribut lain yang tidak bersifat wajib ditulis pada ERD. Seperti Usia, selisih waktu, kelas dan lain sebagainya[30].

2.2.17.3 Relasi

Relasi adalah sebuah hubungan antara beberapa jenis entitas yang berasal dari himpunan entitas-entitas yang berbeda. Relasi ini dilambangkan dengan bentuk ketupat[30]. Dalam ERD relasi yang digunakan ada tiga, yaitu :

2.2.17.3.1 One to One

One to one berarti setiap entitas hanya boleh memiliki relasi dengan satu entitas yang lain. Seperti mahasiswa dengan data NIM, satu mahasiswa hanya memiliki satu NIM[30].

2.2.17.3.2 One to Many

One to many adalah hubungan antara satu entitas dengan beberapa entitas dan sebaliknya. Seperti sekolah dengan siswa, Sekolah memiliki beberapa siswa, dan siswa hanya memiliki satu sekolah[30].

2.2.17.3.3 Many to Many

Many to many adalah hubungan antara beberapa entitas yang memiliki lebih dari suatu relasi. Seperti kelas memiliki beberapa stop kontak, dan beberapa stop kontak dimiliki beberapa ruangan[30].

2.2.17.3.4 Garis

Garis berfungsi untuk menghubungkan antar atribut sebagai bentuk hubungan entitas dari diagram ERD itu sendiri[30].

BAB III

METODOLOGI PENELITIAN

Dalam mengembangkan penelitian ini penulis menggunakan metode *Waterfall*. Metode ini berkembang secara sistematis dari satu tahap ke tahap lainnya seperti air terjun. Sehingga diperlukannya penyelesaian pada setiap tahapan secara berurutan untuk melanjutkan ke tahapan selanjutnya.

3.1. Analisis Pengembangan Perangkat Lunak

Dalam tahapan analisa ini bertujuan untuk menganalisa kebutuhan yang diperlukan untuk merancang aplikasi pada penelitian ini, baik merupa studi pustaka dengan mengambil referensi dari jurnal dan paper di internet, maupun studi lapangan dengan wawancara, dan analisa objek secara langsung.

Analisis yang digunakan pada tahapan ini adalah studi pustaka, studi lapangan.

3.1.1 Studi Pustaka

Pada studi pustaka, dilakukan pencarian metode yang digunakan di jurnal dan artikel pada internet, dalam jangkauan nasional maupun internasional.

Studi yang dilakukan adalah memahami bagaimana penggunaan dan penerapan metode geocode pada aplikasi Android.

3.1.2 Studi Lapangan

Pada Studi Lapangan, dilakukan wawancara pada pemilik dan kasir restoran dan dilakukan pengumpulan data pada lokasi berupa observasi lokasi geologi objek, observasi dan mapping lokasi meja.

3.1.3 Observasi Lokasi Geologi Objek

Berdasarkan lokasi objek pada *Google Map*, lokasi restoran ini lebih masuk ke wilayah perdesaan dibanding dari perkotaan. Menurut lokasi perdesaan yang tidak memiliki banyak gedung-gedung dan bangunan tinggi memiliki keakurasian GPS lebih baik dibanding lokasi perkotaan namun tidak menutup kemungkinan dengan adanya keberadaan pepohonan yang tinggi juga dapat mengganggu dalam keakurasian sinyal GPS.

Dikarenakan kekuatan sinyal satelit GPS tergantung pada adanya interferensi oleh pepohonan besar atau bangunan-bangunan tinggi seperti gedung, lokasi objek tidak terlalu banyak dikelilingi oleh pohon-pohon besar yang dapat mengganggu kekuatan akurasi dari sinyal satelit GPS yang akan digunakan dalam penelitian ini.

Namun keakurasian sinyal GPS juga tidak hanya terpengaruhi oleh banyaknya inteferensi gedung-gedung atau pohon besar, kekuatan sinyal pada *smartphone* juga dapat mempengaruhi pendeteksian lokasi.

Penulis telah melakukan observasi keakurasian GPS dengan menggunakan kartu SIM Axis pada cuaca yang sedang badai hujan pada lokasi objek, cuaca tersebut sangat mempengaruhi dalam keakurasian GPS yang digunakan.

Salah satu diantaranya adalah, posisi GPS yang sering melompat – lompat koordinat, sehingga mempengaruhi dalam keakurasian pendeteksian lokasi dimana *user* berada.

3.1.4 Perubahan Koordinat *Latitude* dan *Longitude* Berdasarkan Pergerakan Lempeng Bumi

Setelah dilakukan observasi kooridnat pada objek selama beberapa hari pada hari yang berbeda, ternyata koordinat yang dideteksi mengalami perubahan, maupun pada *Longitude* atau *Latitude*. Menurut [31] perubahan ini dikarenakan adanya pergerakan lempeng bumi yang selalu bergerak sehingga menimbulkan perubahan titik *Longitude* dan *Latitude* pada suatu tempat.

Hal ini sangat berpengaruh kepada pendeteksian GPS yang menggunakan satelit sebagai metode pengambilan data. Maka dari itu koordinat dapat menjadi kadarluarsa jika tidak di perbarui. Metode untuk memperbarui koordinat yaitu dengan melakukan mapping ulang.

Google Maps selalu melakukan update koordinat setiap tahun untuk menjaga keakuratan dalam mendeteksi koordinat. Google melakukan mapping ulang dengan menggunakan kendaraan google untuk memperbarui koordinat yang dilalui oleh kendaraan tersebut, kendaraan Google juga memiliki kamera 360 derajat diatas atap utuk memperbarui tampilan dari *street view* yang sudah kadarluarsa.

Menurut [32] perubahan ini disebabkan oleh lempengan bumi bernama *lithosphere* yang terpecah menjadi beberapa lempengan kecil yang selalu bergerak kearah yang berbeda-beda dengan kecepatan 50 sampai 100 mm per tahun. Menghasilkan perubahan *longitudinal* sekitar 0.0014 detik busur (arcsec) pertahun.

Jika 1 detik busur (arcsec) adalah 30,87 m maka pergerakan 0.0014 detik busur (arcsec) sebesar

$$arcsecB = \frac{arcsecT}{12}$$

$$arcsecB = \frac{0{,}0014}{12} = 0.00011666666\ arcsec$$

$$arcsecB = Arcsecond\ per\ bulan$$

$$arcsecT = Arcsecond\ per\ tahun$$

Maka pergerakan lempeng bumi perbulan adalah sebesar

$$mB = arcsecB \times 30{,}87$$

$$mB = 0.00011666666 \times 30.87 = 0.00360149979\ m$$

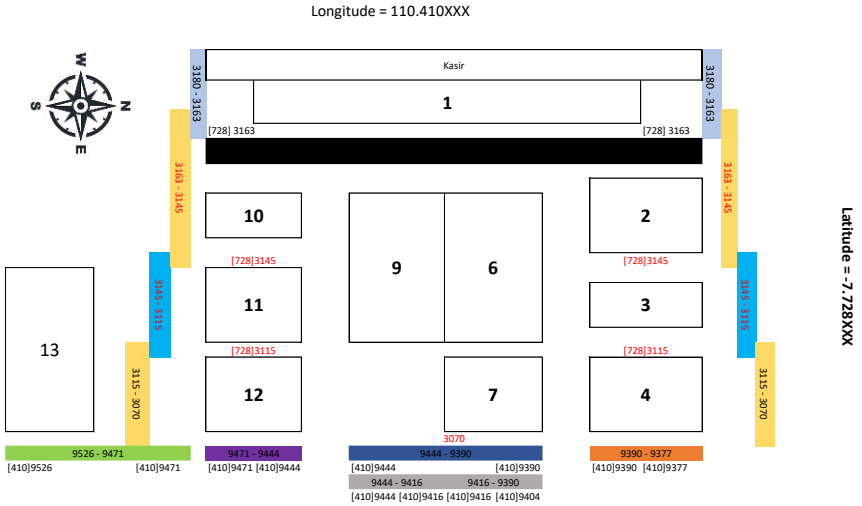
$$mB = Panjang\ pergerakan\ lempeng\ per\ bulan$$

$$arcsecB = Arcsecond\ per\ bulan$$

Jadi setiap 1 bulan pergerakan lempeng bergerak sekitar 0.00360149979 m. Hasil dari perhitungan ini mungkin tidak terlalu akurat dikarenakan pergerakan lempeng bumi akan selalu berubah-ubah menurut kehendak Yang Kuasa.

3.1.5 Mapping Lokasi Meja

Pada tahapan penelitian ini peneliti mengobservasi tiap lokasi meja dan posisi setiap meja dan melakukan mapping meja untuk membuat metode untuk menemukan lokasi *Latitude* dan *Longitude* dan yang dapat mendefinisikan setiap lokasi meja. Mapping ulang lokasi dilakukan seperti pada gambar berikut :



Gambar 7.1 Mapping lokasi tiap meja pada Palgading Resto

Pada Gambar 7.1, setiap meja dikelompokkan berdasarkan *Longitude* (bawah) dan *Latitude* (samping). Semakin ke timur, maka angka *Latitude* semakin bertambah, dan semakin ke selatan maka angka *Longitude* semakin bertambah.

Pada Gambar 7.1, Penulis hanya menuliskan 4 digit belakang tiap lokasi untuk memudahkan dilakukannya analisa.

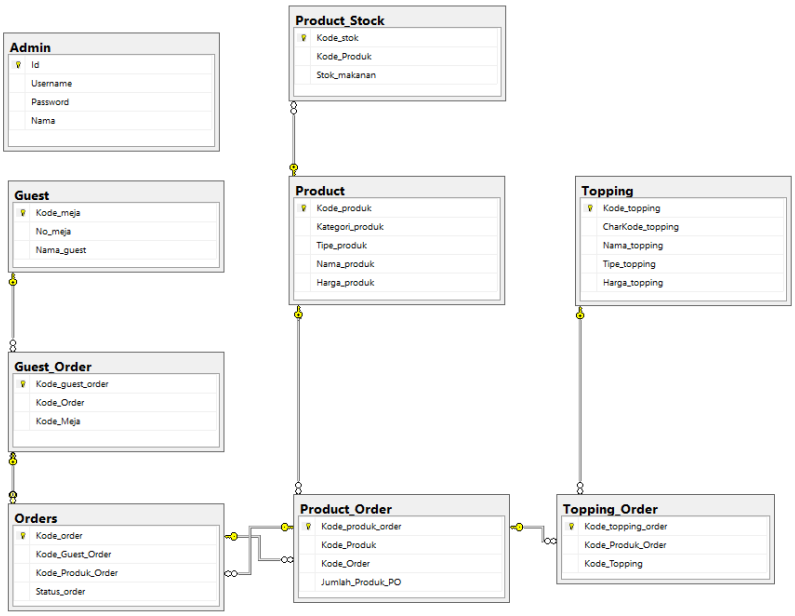
3.2. Desain

Perancangan sistem pada desain ini akan digambarkan menggunakan permodelan basis data dengan menggunakan Rancangan Basis Data yaitu ERD (*Entity Relationship Diagram*) dan UML (*Unified Modeling Language*) yaitu *Use Case Diagram*, *Activity Diagram*, *Sequence Diagram* dan *Class Diagram*.

3.2.1 Rancangan Basis Data

Database merupakan sekelompok file yang berhubungan. Pembuatan Database dilakukan pada *phpmyadmin* lalu di import ke hosting *online*. Database ini bernama db_ppalgading yang berisi beberapa tabel yaitu :

1. Admin
2. Guest
3. Guest_order
4. Orders
5. Product
6. Product_order
7. Product_stock
8. Topping
9. Topping_order



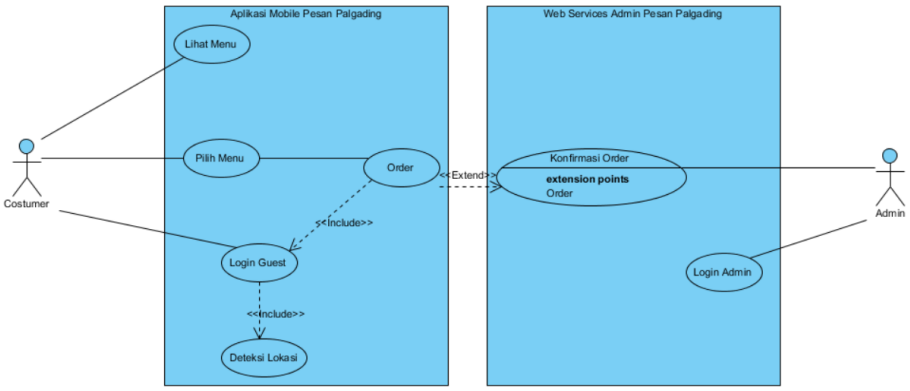
Gambar 7.2 Entity Relationship Diagram

3.2.2 Unified Modeling Language (UML)

UML merupakan sebuah standar penulisan yang berisi bisnis-bisnis proses, penulisan kelas-kelas dalam sebuah bahasa yang lebih spesifik dan detil.

3.2.2.1 Use Case Diagram

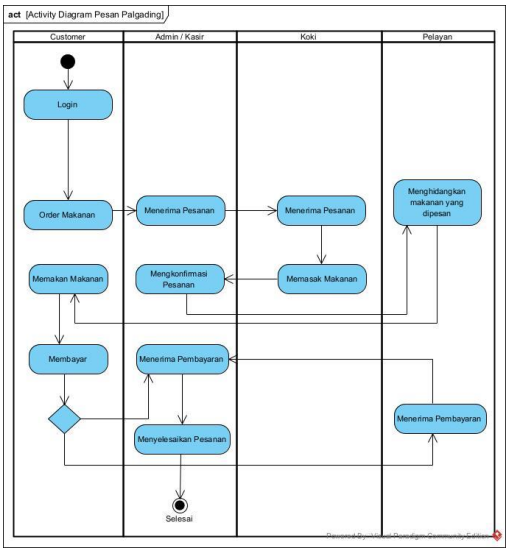
Use Case Diagram mendeskripsikan tipe-tipe interaksi antara pengguna sistem dengan sistem yang digunakan.



Gambar 7.3 Use Case Diagram

3.2.2.2 Activity Diagram

Activity Diagram menggambarkan bagaimana proses-proses yang terjadi pada sistem.

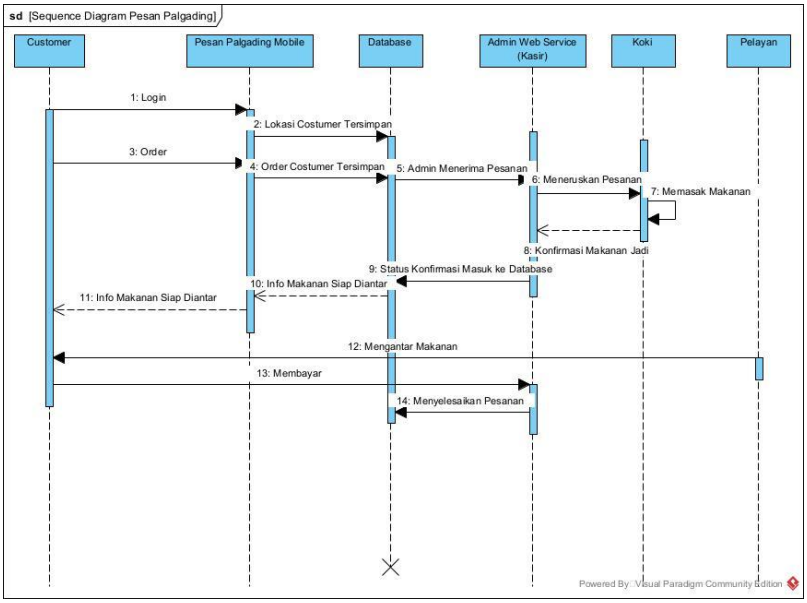


Gambar 7.4 Activity Diagram

3.2.2.3 Sequence Diagram

Sequence Diagram menjelaskan interaksi antar objek berdasarkan urutan waktu.

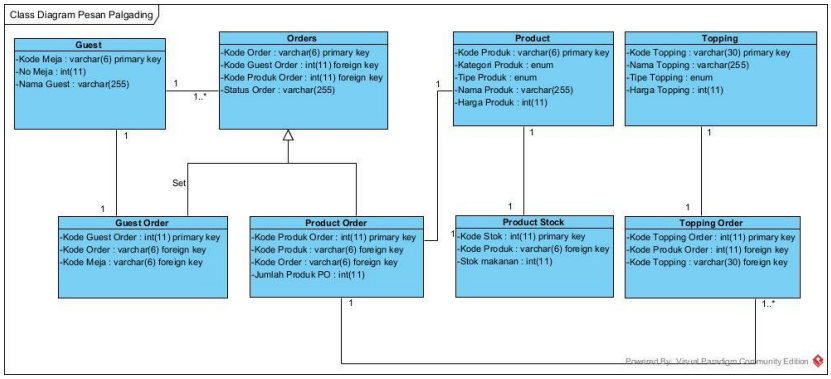
Sequence Diagram juga digunakan untuk menggambarkan urutan atau tahapan yang harus dilakukan untuk mendapatkan suatu output atau hasil pada *use case diagram.*



Gambar 7.5 Sequence Diagram

3.2.2.4 Class Diagram

Class Diagram digunakan untuk memberikan sebuah gambaran mengenai sistem maupun relasi yang terdapat pada sistem yang digunakan.



Gambar 7.6 Class Diagram

BAB IV

IMPLEMENTASI

Dalam Bab ini, penulis menyusun implementasi dari metode yang digunakan.

1.5.1 **Implementasi**

4.1.1 Implementasi Metode Reverse Geocoding

Pada implementasi metode ini penulis membuat logika dalam menentukan lokasi *customer* menempati suatu meja yang telah di mapping lokasinya.

Pertama kali sistem akan mengecek *Longitude* dimana *customer* duduk lalu dicocokkan dengan data yang sudah diterapkan lalu dibandingkan dengan *Latitude* dimana *customer* duduk.

Contohnya, jika *customer* sedang berada pada *Longitude* 110.4109456 dan *Latitude* -7.7283165. Sistem akan mendeteksi *costumer* berada di baris meja 10, 11 dan 12. Lalu sistem akan mengecek kembali dimana *Latitude* tempat *costumer* berada. Sistem akan mendeteksi bahwa *customer* berada di barisan meja 10. Maka ditemukanlah lokasi costumer berada.

Kelemahan pada sistem ini adalah ketika ada meja yang berdekatan memiliki kemungkinan lokasi akan tertukar mengingat pendeteksian lokasi koordinat pada *Google Map* sering melompat-lompat.

Berikut pengimplementasian metode dalam source code pada aplikasi Android :

```
//////////////////////
lat1 = -7.7283180;
lat2 = -7.7283163;
lat3 = -7.7283145;
lat4 = -7.7283115;
lat5 = -7.7283070;

long1 = 110.4109377;
long2 = 110.4109390;
long3 = 110.4109416;
long4 = 110.4109444;
long5 = 110.4109471;
long6 = 110.4109526;
//////////////////////
poslong1 = false;
poslong2 = false;
poslong3 = false;
poslong4 = false;
poslong5 = false;
poslong6 = false;
poslat1 = false;
poslat2 = false;
poslat3 = false;
poslat4 = false;
poslat5 = false;
```

```
posk1confirm = false;
posk2confirm = false;
posk3confirm = false;
posk4confirm = false;
posk6confirm = false;
posk7confirm = false;
posk9confirm = false;
posk10confirm = false;
posk11confirm = false;
posk12confirm = false;
posk13confirm = false;

///kursi
///// kursi 1 2 3 4  ////
if (longitude > long1 && longitude < long2){
    if (latitude > lat1 && latitude < lat2){
        posk1confirm = true;
        poslat1 = true;
    }
    else if (latitude > lat2 && latitude <
lat3){
        posk2confirm = true;
        poslat3 = true;
    }
    else if (latitude > lat3 && latitude <
lat4){
        posk3confirm = true;
        poslat4 = true;
    }
    else if (latitude > lat4 && latitude <
lat5){
        posk4confirm = true;
        poslat5 = true;
    }
    else {
        posk2confirm = false;
        posk3confirm = false;
        posk4confirm = false;
        poslat3 = false;
        poslat4 = false;
        poslat5 = false;
    }
    poslong2 = true;
}
```

```
///// kursi 1 6 7 ////
else if (longitude > long2 && longitude < lon
    if (latitude > lat1 && latitude < lat2){
        posk1confirm = true;
        poslat1 = true;
    }
    else if (latitude > lat3 && latitude <
lat4){
        posk6confirm = true;
        poslat4 = true;
    }
    else if (latitude > lat4 && latitude <
lat5) {
        posk7confirm = true;
        poslat5 = true;
    }
    else {
        posk6confirm = false;
        posk7confirm = false;
        poslat4 = false;
        poslat5 = false;
    }
    poslong3 = true;
      }

//// kursi 1 9 ///
else if (longitude > long3 && longitude < l
    if (latitude > lat1 && latitude < lat2){
        posk1confirm = true;
        poslat1 = true;
    }
    else if (latitude > lat3 && latitude <
lat4){
        posk9confirm = true;
        poslat4 = true;
    }
    else{
        posk9confirm = false;
        poslat4 = false;
    }
    poslong4 = true;
      }
```

```
//// kursi 1 10 11 12 //
else if (longitude > long4 && longitude < l
    if (latitude > lat1 && latitude < lat2){
        posk1confirm = true;
        poslat1 = true;
    }
    else if (latitude > lat2 && latitude <
lat3){
        posk10confirm = true;
        poslat3 = true;
    }
    else if (latitude > lat3 && latitude <
lat4){
        posk11confirm = true;
        poslat4 = true;
    }
    else if (latitude > lat4 && latitude <
lat5){
        posk12confirm = true;
        poslat5 = true;
    }
    else {
        posk10confirm = false;
        posk11confirm = false;
        posk12confirm = false;
        poslat3 = false;
        poslat4 = false;
        poslat5 = false;
    }
    poslong5 = true;
      }

// Kursi 13 //
else if (longitude > long5 && longitude <
long6){
    if (latitude > lat4 && latitude < lat5){
        posk13confirm = true;
        poslat5 = true;
    }
      }
```

```
else {
    poslong1 = false;
    poslong2 = false;
    poslong3 = false;
    poslong4 = false;
    poslong5 = false;
    poslong6 = false;
    poslat1 = false;
    poslat2 = false;
    poslat3 = false;
    poslat4 = false;
    poslat5 = false;
    posk1confirm = false;
    posk2confirm = false;
    posk3confirm = false;
    posk4confirm = false;
    posk6confirm = false;
    posk7confirm = false;
    posk9confirm = false;
    posk10confirm = false;
    posk11confirm = false;
    posk12confirm = false;
    posk13confirm = false;
      }
```

```
if (posk1confirm)
{
    TvLokasiMeja.setText("1");
}
else if (posk2confirm)
{
    TvLokasiMeja.setText("2");
}
else if (posk3confirm)
{
    TvLokasiMeja.setText("3");
}
else if (posk4confirm)
{
    TvLokasiMeja.setText("4");
}
else if (posk6confirm)
{
    TvLokasiMeja.setText("6");
}
else if (posk7confirm)
{
    TvLokasiMeja.setText("7");
}
else if (posk9confirm)
{
    TvLokasiMeja.setText("9");
}
else if (posk10confirm)
{
    TvLokasiMeja.setText("10");
}
else if (posk11confirm)
{
    TvLokasiMeja.setText("11");
}
else if (posk12confirm)
{
    TvLokasiMeja.setText("12");
}
else if (posk13confirm){
    TvLokasiMeja.setText("13");
}
else
{
    TvLokasiMeja.setText("Deteksi Gagal");
    TvLokasiMeja.setTextColor(Color.RED);
      }
```

4.1.2 Implementasi Rancangan Antar Muka

1.5.2 **Pengujian**

Setelah tahapan desain dan tahapan implementasi selesai, maka dilakukan pengujian atu testing program aplikasi untuk melihat apakah sudah sesuai dengan apa yang direncanakan dan dibutuhkan baik pada *input* maupun *output* yang dihasilkan.

Untuk detailnya, pengujian ini menggunakan metode *black box testing* dan metode *white box testing.* Metode *black box testing* ini menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode pada program. *Black box testing* ini menguji fungsi dasar pada aplikasi yang dikembangkan sehingga dapat diketahui apakah sistem berlaku sesuai keinginan user.

Metode *white box testing* ini menguji perangkat apakah fungsi-fungsi *backend* sudah diterapkan dengan benar dan berfungsi dengan baik sehingga berjalan sesuai apa yang direncanakan dan diinginkan.

4.2.1 Black Box Testing

Table 1. Hasil Pengujian Halaman Login *Costumer* Aplikasi Android

No.	Skenario pengujian	*Test case*	Hasil yang diharapkan	Hasil pengujian	Kesimpulan
1	Nama tidak diisi, deteksi meja gagal lalu ditekan login	Nama : kosong GPS : mati	Sistem akan menolak login	Sistem menolak login	Logika vertifikasi user telah berhasil berjalan.
2	Nama diisi, deteksi meja gagal, lalu ditekan login	Nama : Rafly GPS : mati	Sistem akan menolak login	Sistem menolak login	Logika vertifikasi user telah berhasil berjalan.

3	Nama diisi, deteksi meja otomatis berhasil	Nama : Rafly GPS : hidup Deteksi meja (otomatis) : meja 1	Sistem memperbolehkan *costumer* login. Data user tersimpan pada *database*.	Sistem tidak dapat mendeteksi meja 1	Koordinat pada meja 1 telah berubah dikarenakan pergerakan geologis
4	Nama diisi, deteksi meja manual	Nama : Rafly GPS : hidup Deteksi meja (manual) : meja 2	Sistem memperbolehkan *costumer* login. Data user tersimpan pada *database*.	*costumer* dapat login	*costumer* login dan data *costumer* tersimpan dalam database
5	*Costumer* menekan “Lihat menu” pada	*Costumer* menekan button “Lihat menu”	Sistem akan memperlihatkan semua menu yang tersedia	Sistem memperlihatkan menu yang tersedia	Sistem berhasil menampilkan menu yang tersedia

	menu login				

Table 2. Hasil Pengujian Setelah *Costumer* Login ke Dalam Aplikasi Android

No.	Skenario pengujian	*Test case*	Hasil yang diharapkan	Hasil pengujian	Kesimpulan
1	*Costumer* menekan menu profil	*Costumer* telah berhasil masuk ke dalam aplikasi	Nama, kode kursi & no kursi terdeteksi dan ditampilkan dengan benar pada menu profil	Nama, kode kursi dan no kursi terdeteksi dan ditampilkan	Sistem telah berhasil menyimpan dan menampilkan data *costumer*

Table 3. Hasil Pengujian Ketika *Costumer* Melakukan Order Pada Aplikasi Android

No.	Skenario pengujian	*Test case*	Hasil yang diharapkan	Hasil pengujian	Kesimpulan
1	*Costumer* memilih	Makanan : Mie	*Costumer* akan disuruh untuk	Pesanan tidak dapat dibuat	Sistem mencegah *costumer* untuk

	makanan untuk diorder	Ayam Original	memilih tipe makanan terlebih dahulu	sebelum *costumer* memilih tipe makanan terlebih dahulu	melakukan pesanan tanpa mendeskripsikan tipe makanan yang akan dipesan
2	*Customer* menambahkan jumlah makanan lalu menekan konfirmasi pesanan	Makanan : Mie Ayam Original Tipe Makanan : Kuah Jumlah makanan : 2	Sistem akan menghitung jumlah harga dari 2 Mie Ayam Original	Sistem menghitung jumlah harga dari pesanan yang *costumer* pesan	Sistem telah berhasil menghitung jumlah biaya dari pesanan yang dipilih
3	*Customer* tidak memilih tipe makanan lalu menekan konfirmasi pesanan	*Radio Button* tipe makanan tidak terpilih satupun	Sistem akan menolak melakukan perhitungan total pesanan sebelum	Sistem menolak untuk melakukan perhitungan total pesanan	Sistem akan menyuruh *costumer* untuk memilih tipe makanan terlebih dahulu

			customer memilih tipe makanan		
4	*Customer* memilih satu atau lebih topping lalu menekan konfirmasi pesanan	Makanan : Mie Ayam Original Tipe Makanan : Kuah Topping : Extra Sawi Tombol konfirmasi pesanan ditekan	Sistem akan menghitung total harga dari jumlah makanan dan harga topping yang dipilih	Sistem menghitung total jumlah harga dari makanan dan topping yang dipilih	Sistem telah berhasil menghitung total biaya yang harus dibayar *costumer*
5	*Customer* melakukan order	Makanan :	Sistem akan menyimpan order kedalam	Sistem menyimpan order kedalam	Sistem telah berhasil menginput orderan

		Mie Ayam Original Tipe Makanan : Kuah Topping : Extra Sawi *Customer* menekan tombol order	*database* dan melanjutkan ke halaman notifikasi dan menampilkan order yang dibuat dan status pesanan.	database dan melanjutkan ke halaman notifikasi menampilkan order dan status pesanan yang dibuat	*costumer* ke database dan menampilkan pesanan *costumer* pada aplikasi

Table 4. Hasil Pengujian Halaman Login Admin pada Web Admin

No.	Skenario pengujian	*Test case*	Hasil yang diharapkan	Hasil pengujian	Kesimpulan
1	*Username* dan *Password* tidak	*Username* : kosong *Password* : kosong	Sistem akan menolak untuk login	Sistem menolak login	Sistem menolak login dikarenakan *username* dan

	diisi lalu login				*password* kosong
2	*Username* tidak diisi dan *Password* diisi lalu login	*Username* : kosong *Password* : admin123	Sistem akan menolak untuk login.	Sistem menolak login	Sistem menolak login dikarenakan *username* kosong
3	*Username* diisi dan *Password* tidak diisi lalu login	*Username* : Admin *Password* : kosong	Sistem akan menolak untuk login.	Sistem menolak login	Sistem menolak login dikarenakan *password* kosong
4	*Username* diisi dan *Password* diisi dengan benar lalu login	*Username* : Admin *Password* : Admin123	Sistem akan memperbolehkan admin untuk masuk	Sistem memperbolehkan login	Sistem berhasil memvertifikasi *Admin*.

Table 5. Hasil Pengujian Halaman *Dashboard* Admin pada Web Admin

No.	Skenario pengujian	*Test case*	Hasil yang diharapkan	Hasil pengujian	Kesimpulan
1	Website melakukan *auto refresh* tiap 10 detik	Admin telah login ke dalam dashboard	Sistem melakukan *auto refresh* setiap 10 detik	Sistem *refresh* otomatis setiap 10 detik	Fungsi *auto refresh* berjalan dengan sempurna
2	Admin memberhentikan *auto refresh* dengan menekan tombol *stop refresh*	Admin menekan tombol *stop refresh*	Sistem memberhentikan *auto refresh* dan web tidak akan *refresh* otomatis	Web berhenti *refresh* automatis	Sistem menghentikan fungsi auto refresh
3	Saat ada order masuk, web akan mengeluarkan suara	Web mengeluarkan suara berulang ulang	Web mengeluarkan suara notifikasi menandakan	Web mengeluarkan suara ketika ada	Sistem telah berhasil mendeteksi inputan baru dari

		sampai web ter refresh	pesanan masuk	pesanan masuk	database lalu mengeluarkan suara notifikasi
4	Saat ada order masuk, order akan ditampilkan	Order ditampilkan pada tabel pesanan baru dan pada tabel lokasi meja	Pada tabel pesanan baru, order ditampilkan secara sederhana. Pada tabel lokasi meja, kotak tabel akan berwarna hijau tergantung pada no meja berapa orderan tersebut	Tabel pesanan baru ditampilkan pesanan yang baru saja masuk berwarna hijau	Sistem telah berhasil mendeteksi inputan baru pada table orders dan ditampilkan pada web
5	Admin menekan	Admin menekan	Tabel detail order akan	Tabel detail order	Sistem telah mengeksek

	order pada tabel pesanan baru	order pada pesanan baru	menampilkan isi dari semua order berdasarkan no meja pada order tersebut	ditampilkan dan berisi dari semua pesanan berdasarkan no meja	usi query yang mendeteksi inputan pesanan berdasarkan nomor yang ditekan
6	Admin menekan no meja yang berwarna hijau pada tabel lokasi meja	Admin menekan no meja yang berwarna hijau pada tabel lokasi meja.	Tabel detail order akan menampilkan isi dari semua order berdasarkan no meja pada order tersebut.	Tabel detail order ditampilkan dan berisi dari semua pesanan berdasarkan no meja	Sistem telah mengeksek usi query yang mendeteksi inputan pesanan berdasarkan nomor yang ditekan
7	Admin menekan no meja berwarna putih pada	Admin menekan no meja yang berwarna putih pada	Tabel detail order tidak akan menampilkan order.	Tabel detail order tidak akan menampilk an apapun	Sistem akan mendeteksi inputan pesanan berdasarkan

	tabel lokasi meja	tabel lokasi meja			nomor yang ditekan, jika tidak ada inputan atau pesanan, maka table detail order tidak akan muncul apa - apa
8	Admin menekan tombol konfirmasi pada tabel detail order kolom konfirmasi pesanan	Tombol konfirmasi pada tabel detail order kolom konfirmasi pesanan ditekan	Status pesanan akan berubah menjadi Belum Dibayar dan baris data akan berubah menjadi warna kuning pada tabel detail order menandakan	Status pesanan berubah menjadi Belum Dibayar dan baris data pada detail order akan berwarna kuning	Sistem telah berhasil mengubah status pesanan menjadi Belum Dibayar

			pesanan telah dikonfirmasi namum belum dibayar		
9	Admin menekan tombol Selesai pada tabel detail order kolom konfirmasi pesanan	Tombol Selesai pada tabel detail order kolom konfirmasi pesanan ditekan	Status pesanan akan berubah menjadi Selesai dan baris data akan menghilang pada tabel detail order menandakan pesanan telah selesai setelah *consumer* membayar	Status pesanan berubah menjadi Selesai dan menghilang pada detail order	Sistem telah berhasil mengubah status pesanan menjadi Selesai dan tidak akan menampilkan pesanan pada table detail order

Table 6. Hasil Pengujian Integrasi Web Service ke Aplikasi Android

No.	Skenario pengujian	*Test case*	Hasil yang diharapkan	Hasil pengujian	Kesimpulan
1	Admin mengonfirmasi pesanan lewat web	Tombol konfirmasi ditekan oleh admin pada web	Status pesanan pada tab notifikasi pada aplikasi android berubah menjadi Belum Dibayar menandakan pesanan telah dikonfirmasi dan akan segera diantar ke *consumer*	Status pesanan pada halaman notifikasi aplikasi android berubah menjadi Belum Dibayar	Sistem telah berhasil mengubah status pesanan menjadi Belum dibayar dan ditampilkan pada aplikasi android
2	Admin menyelesaikan pesanan lewat web	Tombol selesai ditekan oleh admin pada web	Status pesanan pada tab notifikasi pada aplikasi android berubah menjadi selesai menandakan pesanan telah selesai dan akan	Status pesanan pada halaman notifikasi aplikasi android berubah	Sistem telah berhasil mengubah status pesanan menjadi Selesai dan ditampilkan

			diantar ke *costumer*	menjadi Selesai	pada aplikasi android, namun tidak akan dihitung pada total bayar dikarenakan pesanan telah dibayar

4.2.2 White Box Testing

Table 7. Hasil Pengujian Koordinat pada Aplikasi Android

No.	Skenario pengujian	*Test case*	Hasil yang diharapkan	Hasil pengujian	Kesimpulan
1	Dilakukan pengetesan koordinat setelah beberapa bulan pengambilan data	Test koordinat di objek penelitian	Koordinat tidak berubah dari pendeteksian pada waktu pengambilan data	Koordinat berubah	Koordinat berubah dikarenakan pergerakan geologis.

Table 8. Hasil Pengujian Backend pada Aplikasi Android

No.	Skenario pengujian	*Test case*	Hasil yang diharapkan	Hasil pengujian	Kesimpulan
1	Logika berhasil menemukan lokasi *costumer* berada berdasarkan koordinat	Auto deteksi berjalan dan menampilkan lokasi meja	Sistem berhasil menemukan lokasi user dengan benar	Sistem gagal menemukan lokasi user dengan GPS	Koordinat yang dimasukan pada aplikasi telah *expire* dikarenakan koordinat

	yang diterapkan				pada objek telah berubah dikarenakan pergerakan geologis.
2	Order telah dibuat oleh *customer*	*Costumer* membuat order	Order tersimpan pada database di tabel orders	Order telah tersimpan pada database di table orders	Sistem telah berhasil menyimpan order ke dalam databse setalah *costumer* membuat order
3	*Costumer* login	*Costumer* login dengan berhasil	Data *Costumer* tersimpan pada database di tabel guest	Data *Costumer* tersimpan pada database di tabel guest	Data *Costumer* telah tersimpan pada database di table guest

					setelah *costumer* login
4	Data lokasi *Costumer* ditampilkan	Data *Costumer* ditampilkan	Data *Costumer* ditampilkan pada tab profile	Data *Costumer* ditampilkan pada tab profile	Sistem telah berhasil mengambil data *costumer* pada database dan ditampilkan
5	Data order *Costumer* ditampilkan	Data order *Costumer* ditampilkan	Data semua order yang dibuat oleh user ditampilkan pada tab notifikasi	Data semua order yang dibuat oleh user ditampilkan pada tab notifikasi	Sistem telah berhasil mengambil data order yang dibuat user dari database dan ditampilkan pada

					aplikasi android

No.	Skenario pengujian	*Test case*	Hasil yang diharapkan	Hasil pengujian	Kesimpulan
1	Dashboard dapat menyesuaikan ukuran jika dibuka melalui *smartphone*	Web dibuka pada *smartphone*	Layout web menjadi responsif jika dibuka melalui *smartphone*	Layout web menjadi responsif jika dibuka melalui *smartphone*	Sistem telah berhasil menerapkan layout responsive ketika dibuka dengan *smarphone*
2	Web cepat dalam menampilkan dashboard	Web memuat dashboard	Web memuat dashboard dengan cepat	Web memuat dashboard dengan cepat	Web telah dibuat seringan mungkin agar mempermudah admin dalam mengakses web dan menjalankannya

1.5.3 **Pendukung (*support*) atau Pemeliharaan (*maintenance*)**

Dalam perancangan dan pembuatan software, software tidak selalu sempurna dan berjalan semestinya, mungkin masih ada beberapa *error* dan *bug* yang berada dalam aplikasi yang tidak terdeteksi pada tahap pengujian.

Dan pada tahapan pemeliharaan, perangkat lunak harus beradaptasi dengan lingkungan baru, sehingga pemeliharaan harus dilakukan agar aplikasi dapat berjalan semestinya, dengan tidak perlu membuat aplikasi baru, namun hanya perlu untuk dilakukan pengembangan pada aplikasi yang sudah ada.

Dikarenakan koordinat yang selalu berubah-ubah maka aplikasi perlu dilakukan pembaruan terhadap koordinat yang baru agar dapat akurat dalam mendeteksi lokasi pemesan makanan secara otomatis.

BAB V

PENUTUP

5.1 Kesimpulan

Setelah dilakukannya pengujian aplikasi pada objek stelah beberapa bulan pengerjaan aplikasi, ternyata koordinat yang dideteksi berubah, dikarenakan lempengan bumi akan terus bergerak sehingga menimbulkan nya perubahan pada *Longitude* dan *Latitude* pada permukaan bumi.

Sementara itu metode *Geocode* ini berbasis *GPS* yang mengandalkan sistem pendeteksian koordinat menggunakan satelit di luar bumi yang hanya mendeteksi permukaan bumi secara langsung, sehingga satelit tidak dapat menyesuaikan koordinat tanpa dilakukannya pembaruan manual oleh manusia itu sendiri, *Google Maps* juga harus memperbarui peta digital mereka setiap tahun untuk mengganti koordinat yang telah berubah dari waktu ke waktu.

Perubahan koordinat ini juga tidak selalu sama setiap tahun sehingga tidak dapat dibuat sebuah alat yang dapat mendeteksi secara otomatis kapan koordinat akan berubah dan berapa perubahan koordinat yang akan terjadi.

Dikarenakan penelitian ini menggunakan metode *Geocode* yang mengandalkan *GPS* dalam rangkap ruang yang kecil, metode ini tidak akan berjalan dengan efektif jika digunakan dengan aplikasi yang membutuhkan jangka waktu yang panjang dalam pendeteksian koordinat nya seperti aplikasi pemesanan makanan ini.

Selain ruang lingkup yang terlalu kecil (hanya sebatas restoran), koordinat juga akan selalu berubah ubah tergantung dengan lempeng bumi yang selalu bergerak, ditambah lagi jika terjadi gempa bumi pada suatu daerah tersebut, lokasi tersebut akan berubah jauh koordinat nya.

5.2 Saran

Berdasarkan simpulan yang telah dipaparkan diatas maka dalam kesempatan ini penulis akan menyampaikan beberapa saran yaitu sebagai berikut:

1. Bagi Penelitian Lanjutan

Pada penggunaan metode *Geolocation* ini tidak direkomendasikan untuk diterapkan pada aplikasi yang memiliki ruang lingkup objek yang kecil, yang dimaksud kecil adalah dalam ruang lingkup rumah makan, restoran, foodcourt, café. Dikarenakan keakurasian dalam pendeteksian koordinat akan sangat rendah walaupun lokasi objek terdapat pada lokasi yang sepi tanpa ada bangunan bangunan tinggi.

Metode ini juga tidak disarankan untuk digunakan pada jangka panjang dikarenakan koordinat yang selalu berubah ubah dikarenakan pergerakan lempeng bumi yang selalu bergerak sehingga perlu dilakukan pembaruan koordinat secara berkala jika ingin pendeteksian koordinat secara akurat.

DAFTAR PUSTAKA

[1] 3201412039 Alfian Adestya Putra, "PEMANFAATAN APLIKASI GOOGLE MAPS PADA SMARTPHONE ANDROID SEBAGAI SARANA BELAJAR NAVIGASI MAHASISWA FAKULTAS ILMU SOSIAL UNIVERSITAS NEGERI SEMARANG," other, Universitas Negeri Semarang, 2016. Accessed: Jan. 21, 2021. [Online]. Available: http://lib.unnes.ac.id/27312/

[2] R. Limia Budiarti and W. Adriana, "Pemanfaatan Google Maps API dalam Pemetaan dan Pemberdayaan Pariwisata Desa Di Indonesia Berbasis Web-Mobile," *Indones. J. Comput. Sci.*, vol. 8, no. 1, pp. 55–65, Apr. 2019, doi: 10.33022/ijcs.v8i1.163.

[3] A. Muawwal, "Implementasi Teknologi GPS Tracking Smartphone Sebagai Aplikasi Monitoring Lokasi Anak," p. 5.

[4] R. Rismayani, "PEMANFAATAN TEKNOLOGI GOOLE MAPS API UNTUK APLIKASI LAPORAN KRIMINAL BERBASIS ANDROID PADA POLRESTABES MAKASSAR," *J. Penelit. Pos Dan Inform.*, vol. 6, no. 2, p. 185, Dec. 2016, doi: 10.17933/jppi.2016.060205.

[5] A. Ginjala, "Emergency Search Using Android App," 2015, Accessed: Feb. 25, 2021. [Online]. Available: https://library.ndsu.edu/ir/handle/10365/25510

[6] S. Sandheep, H. John, A. Harikumar, and J. V. Panicker, "BusTimer: An android based application for generating bus schedules using crowdsourcing," in *2017 International Conference on Technological Advancements in Power and Energy (TAP Energy)*, Dec. 2017, pp. 1–6. doi: 10.1109/TAPENERGY.2017.8397270.

[7] R. Ariyanti and I. Kanedi, "PEMANFAATAN GOOGLE MAPS API PADA SISTEM INFORMASI GEOGRAFIS DIREKTORI PERGURUAN TINGGI DI KOTA BENGKULU," vol. 11, no. 2, p. 11, 2015.

[8] S. McQuire, "One map to rule them all? Google Maps as digital technical object," *Commun. Public*, vol. 4, no. 2, pp. 150–165, Jun. 2019, doi: 10.1177/2057047319850192.

[9] P. Doshi, P. Jain, and A. Shakwala, "Location Based Services and Integration of Google Maps in Android," *Int. J. Eng. Comput. Sci.*, vol. 3, no. 03, Art. no. 03, Mar. 2014, Accessed: Jan. 22, 2021. [Online]. Available: http://103.53.42.157/index.php/ijecs/article/view/190

[10] S. Alfeno and R. E. C. Devi, "Implementasi Global Positioning System (GPS) dan Location Based Service (LSB) pada Sistem Informasi Kereta Api untuk Wilayah Jabodetabe," *J. SISFOTEK Glob.*, vol. 7, no. 2, Art. no. 2, Sep. 2017, doi: 10.38101/sisfotek.v7i2.146.

[11] Z. Yin, A. Ma, and D. W. Goldberg, "A deep learning approach for rooftop geocoding," *Trans. GIS*, vol. 23, no. 3, pp. 495–514, 2019, doi: https://doi.org/10.1111/tgis.12536.

[12] "What is geocoding?—ArcMap | Documentation." https://desktop.arcgis.com/en/arcmap/latest/manage-data/geocoding/what-is-geocoding.htm (accessed Jan. 22, 2021).

[13] L. Zeigermann, "Opencagegeo: Stata Module for Forward and Reverse Geocoding," p. 10.

[14] "Overview | Geocoding API," *Google Developers*. https://developers.google.com/maps/documentation/geocoding/overview (accessed Jan. 22, 2021).
[15] "GSP 270: Latitude and Longitude." http://gsp.humboldt.edu/OLM/Lessons/GIS/01%20SphericalCoordinates/Latitude_and_Longitude.html (accessed Dec. 17, 2020).
[16] "Understanding Latitude and Longitude." https://journeynorth.org/tm/LongitudeIntro.html (accessed Jan. 22, 2021).
[17] C.-12 Foundation, "Longitude | CK-12 Foundation." https://www.ck12.org/book/physics-from-stargazers-to-starships/section/10.2/ (accessed Jan. 22, 2021).
[18] "Greenwich meridian | geography," *Encyclopedia Britannica*. https://www.britannica.com/place/Greenwich-meridian (accessed Jan. 22, 2021).
[19] J. F. Tompoh, S. R. Sentinuwo, and A. A. E. Sinsuw, "Rancang Bangun Aplikasi Pemesanan Menu Makanan Restoran Berbasis Android," *J. Tek. Inform.*, vol. 9, no. 1, Art. no. 1, Oct. 2016, doi: 10.35793/jti.9.1.2016.13749.
[20] "Android Definition." https://techterms.com/definition/android (accessed Jan. 22, 2021).
[21] "What is android - javatpoint," *www.javatpoint.com*. https://www.javatpoint.com/android-what-where-and-why (accessed Jan. 22, 2021).
[22] M. T. Prihandoyo, "Unified Modeling Language (UML) Model Untuk Pengembangan Sistem Informasi Akademik Berbasis Web," *J. Inform.*, p. 4, 2018.
[23] D. Intern, "Apa itu UML? Beserta Pengertian dan Contohnya," *Dicoding Blog*, May 11, 2021. https://www.dicoding.com/blog/apa-itu-uml/ (accessed Dec. 24, 2021).
[24] "JSON." https://www.json.org/json-en.html (accessed Dec. 24, 2021).
[25] "What is an API? (Application Programming Interface)," *MuleSoft*. https://www.mulesoft.com/resources/api/what-is-an-api (accessed Dec. 24, 2021).
[26] D. S. Agnes, "Memahami API, REST API, dan RESTful API," *wripolinema*, Sep. 17, 2020. https://medium.com/wripolinema/memahami-api-rest-api-dan-restful-api-5fd2327edd3c (accessed Dec. 24, 2021).
[27] "Volley overview," *Android Developers*. https://developer.android.com/training/volley (accessed Dec. 24, 2021).
[28] T. D. Wismarini and A. Prihandono, "RANCANG BANGUN APLIKASI ANDROID TERINTEGRASI WEB SERVICE DENGAN VOLLEY UNTUK LAYANAN PUBLIK," *Dinamik*, vol. 25, no. 1, pp. 10–19, Jun. 2020, doi: 10.35315/dinamik.v25i1.7515.
[29] M. A. Lestari, M. Tabrani, and S. Ayumida, "SISTEM INFORMASI PENGOLAHAN DATA ADMINISTRASI KEPENDUDUKAN PADA KANTOR DESA PUCUNG KARAWANG," vol. 13, no. 3, p. 8, 2018.
[30] "Apa itu ERD? Kenali Jenis, Komponen dan Tools yang Digunakan," *Sekawan Media | Software House & System Integrator Indonesia*, Jan. 04,

2021. https://www.sekawanmedia.co.id/blog/apa-itu-erd/ (accessed Dec. 28, 2021).

[31] F.-B. Mocnik and R. Westerholt, "The effect of tectonic plate motion on georeferenced long-term global datasets," *Int. J. Appl. Earth Obs. Geoinformation*, vol. 94, p. 102183, Feb. 2021, doi: 10.1016/j.jag.2020.102183.

[32] "Longitude," *Wikipedia*. Mar. 15, 2022. Accessed: Mar. 17, 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Longitude&oldid=1077337933